Министерство образования Республики Беларусь
Учреждение Образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Кафедра электронных вычислительных средств

Лабораторная работа № 5
«ИЗУЧЕНИЕ МЕТОДОВ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ ПРИ ОПТИМАЛЬНОМ ПРОЕКТИРОВАНИИ»

Выполнили:
ст. гр. 850702
Маковский Р. А.
Турко В. Д.

Проверил:
Станкевич А. В.

Минск 2020

**ЦЕЛЬ РАБОТЫ:**

Изучить методы решения задачи линейного программирования, используемые при проектировании ЭВС.

**ИСХОДНЫЕ ДАННЫЕ(ВАРИАНТ 3):**

**3.** $F = 2x_1 - 4x_2 + 3 \rightarrow \max$

$$\begin{cases} 3x_1 \geq 8x_2 \\ 2x_1 + 3x_2 \leq 6 \\ x_1 - x_2 \leq 4 \\ -2x_1 + x_2 \leq 2 \end{cases}, \qquad x_i \geq 0$$

**ХОД РАБОТЫ:**

# Init data

```
In[•]:= (*xi≥0*)
InitF := 2 * x1 - 4 * x2 + 3;

(*n<m*)
InitLimit1 := 3 * x1 ≥ 8 * x2;
InitLimit2 := 2 * x1 + 3 * x2 ≤ 6;
InitLimit3 := x1 - x2 ≤ 4;
InitLimit4 := -2 * x1 + x2 ≤ 2;

export[name_, data_] := Export["ММиАвАП\\lab5\\" <> name <> ".png", data]
                        экспортировать
```

# Geometric

## Limits

```
In[•]:= L1 := 3 * x1 - x3 == 8 * x2;
L2 := 2 * x1 + 3 * x2 + x4 == 6;
L3 := x1 - x2 + x5 == 4;
L4 := -2 * x1 + x2 + x6 == 2;
```

## Used functions

```
In[•]:= FormInequalities := Module[{a, b, c, d},
                          программный модуль
    a = (x3 /. Solve[L1, x3]) [[1]] ≥ 0;
               решить уравнения
    b = (x4 /. Solve[L2, x4]) [[1]] ≥ 0;
               решить уравнения
    c = (x5 /. Solve[L3, x5]) [[1]] ≥ 0;
               решить уравнения
    d = (x6 /. Solve[L4, x6]) [[1]] ≥ 0;
               решить уравнения
    Return[{a, b, c, d}];
    вернуть управление
   ];

FormInequalitiesInv := Module[{a, b, c, d},
                       программный модуль
    a = (x3 /. Solve[L1, x3]) [[1]] ≤ 0;
               решить уравнения
    b = (x4 /. Solve[L2, x4]) [[1]] ≤ 0;
               решить уравнения
    c = (x5 /. Solve[L3, x5]) [[1]] ≤ 0;
               решить уравнения
    d = (x6 /. Solve[L4, x6]) [[1]] ≤ 0;
               решить уравнения
```

```
    решить уравнения
  Return[{a, b, c, d}];
  вернуть управление
 ];

FormArea := Module[{inEqs = FormInequalitiesInv, plot},
            программный модуль
  plot = RegionPlot[{inEqs}, {x1, 0, 10}, {x2, 0, 5}, PlotLegends → {FormInequalities},
         визуализация геометрической области на плоскости   легенды графика
    AxesLabel → Automatic, AspectRatio → Automatic];
    обозначения·· автоматиче··· аспектное отн··· автоматический
  Return[plot];
  вернуть управление
 ];

FindMinMax := Module[{min, max, inEqs = FormInequalities},
              программный модуль
  min = FindMinimum[{2 * x1 - 4 * x2 + 3, inEqs[[1]] &&
        найти минимум
      inEqs[[2]] && inEqs[[3]] && inEqs[[4]] && x1 ≥ 0 && x2 ≥ 0}, {x1, x2}];
  max = FindMaximum[{2 * x1 - 4 * x2 + 3, inEqs[[1]] && inEqs[[2]] &&
        найти максимум
      inEqs[[3]] && inEqs[[4]] && x1 ≥ 0 && x2 ≥ 0}, {x1, x2}];
  Return[{min, max}];
  вернуть управление
 ];

ReduceIneqs := Module[{res, inEqs = FormInequalities},
               программный модуль
  res = Reduce[
        привести
    inEqs[[1]] && inEqs[[2]] && inEqs[[3]] && inEqs[[4]] && x1 ≥ 0 && x2 ≥ 0, {x1, x2}];
  Return[res];
  вернуть управление
 ];
```

## Using plots

```
In[•]:= Row[{ReduceIneqs[[1]], ReduceIneqs[[2]], ReduceIneqs[[3]], ReduceIneqs[[4]]}, "\n"]
        ряд
        Module[{a, b, c, d, min, max},
        программный модуль
         OrigF[y_] := x2 /. Solve[InitF == y, x2];
                            решить уравнения
         Show[{Plot[{OrigF[0], OrigF[5], OrigF[10]}, {x1, 0, 5},
         пока··· график функции
            PlotLegends → {InitF == 0, InitF == 5, InitF == 10}, PlotStyle → Thick], FormArea}]
            легенды графика                                   стиль графика жирный
        ]
```

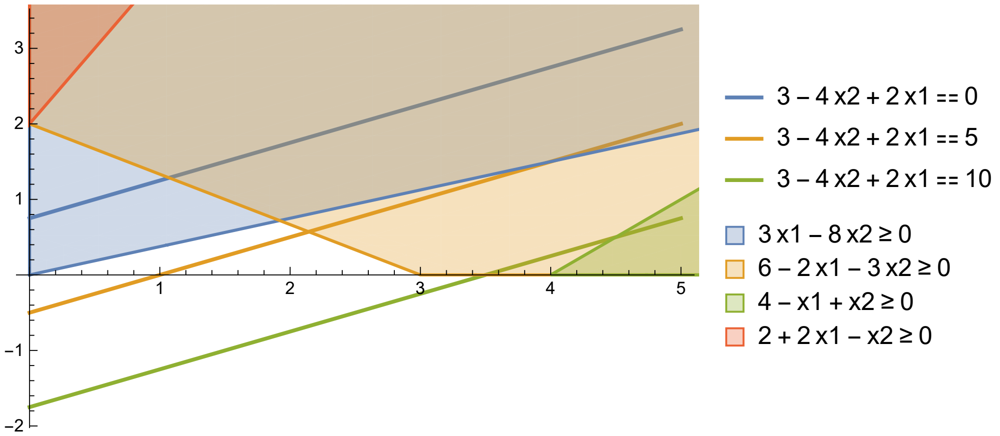
Out[ ]= x1 == 0 && x2 == 0

$0 < x1 \leq \frac{48}{25}$ && $0 \leq x2 \leq \frac{3\,x1}{8}$

$\frac{48}{25} < x1 < 3$ && $0 \leq x2 \leq \frac{1}{3}(6 - 2\,x1)$

x1 == 3 && x2 == 0

Out[ ]=




In[ ]:=
```
Row[
{"Fmin", Row[{Assuming[x1 == 0 && x2 == 0, Refine[InitF]], "(x1→0, x2→0)"}, ","]}, "="]
```
ряд ряд предполагая уточнить

Out[ ]= Fmin = 3, (x1→0, x2→0)

### Built-in functionality (just for test)

In[ ]:=
```
FindMinMax
```

Out[ ]= {{3., {x1 → 0., x2 → 0.}}, {9., {x1 → 3., x2 → 0.}}}

## Simplex

### Limits

In[ ]:=
```
Fs := 3 - (-2 * x1 + 4 * x2);
L1s := x3 == 0 - (-3 * x1 + 8 * x2);
L2s := x4 == 6 - (2 * x1 + 3 * x2);
L3s := x5 == 4 - (1 * x1 - 1 * x2);
L4s := x6 == 2 - (-2 * x1 + 1 * x2);
```

т.к. условие оптимальности не выполняется (все коэффициенты при свободных переменных разных знаков), необходимо выбрать новую базисную переменную

в соответствии с правилом выбора свободной переменной в базис переводим переменную x2

в соответствии с правилом выбора базисной переменной в свободные переменные переводим x3

In[∘]:=
```
Fs := 3 - (-x1 / 2 - x3 / 2);
L1s := x2 == 0 - (-3 * x1 / 8 + x3 / 8);
L2s := x4 == 6 - (2 * x1 + 3 * x2);
L3s := x5 == 4 - (1 * x1 - 1 * x2);
L4s := x6 == 2 - (-2 * x1 + 1 * x2);
```

т.к. теперь коэффициенты при всех свободных переменных целевой функции отрицательны, выполняется условие оптимального решения
и целевая функция будет иметь минимальное значение

In[∘]:=
```
Row[{"Fmin", Row[{Assuming[x1 == 0 && x3 == 0, Refine[Fs]], "(x1→0, x3→0)"}, ","]}, "="]
```
ряд ряд предполагая уточнить

Out[∘]=
```
Fmin = 3, (x1→0, x3→0)
```

**ВЫВОД:**

В лабораторной работе мы познакомились с методами линейного программирования. В частности, мы максимизировали функцию при помощи симплекс-метода и графического метода. Для маленького числа размерностей в постановке линейной задачи может хорошо подойти графический метод, но если число размерностей велико, то решение подобных задач остается за симплекс-методом.